RX02
Diagnostic Manual
Version 2

JUNE 1981 – MICRO TECHNOLOGY, INC.

REVISIONS TO THE RXØ2 DIAGNOSTIC MANUAL

The RX02 diagnostic has been modified to also test the MXV22 controller. The changes are for the most part transparent to the operator. The program version displayed at start-up is Version 2A. Minor changes were made to the test programs and the new listings are in Section 4.

The following changes should be made to this document:

1. Version 2 should be changed to Version 2A.

2. References to the MXV21 should be changed to MXV21/MXV22.

Note that the old version of the diagnostic will not operate with 22-bit addressing enabled and will not test four drives simultaneously.

MARCH 1983 - MICRO TECHNOLOGY, INC.

PREFACE

The Micro Technology, Inc. RX02 diagnostic performs various tests to confirm that the MXV21 controller is functioning properly. Prompts are displayed to assist in selecting which test(s) to operate. Error messages identify detected failures.

PREFACE

The Micro Technology, Inc. RX02 diagnostic performs various tests to confirm that the MXV21 controller is functioning properly. Prompts are displayed to assist in selecting which test(s) to operate. Error messages identify detected failures.

TABLE OF CONTENTS

1.1 STARTING THE DIAGNOSTIC

The diagnostic can be booted from the diagnostic diskette. The diagnostic diskette is in RT-11 format so the diagnostic can also be read in using the RT-11 GET command and started at location 2000, which is the restart address. After the program is started, formatted scratch diskette(s) should be inserted in the drive(s) to be tested, and a carriage return entered to continue the program.

CAUTION

Any diskette inserted may be written on, so the diagnostic diskette should be removed.

1.2 OPERATOR PROMPTS

The prompt "INSERT SCRATCH DISKETTE(S)-TYPE CR TO START" provides the operator a chance to remove the diagnostic diskette. Insert formatted scratch diskettes in the drives to be tested and enter a carriage return. If the test is to be performed on a UNIBUS device type Y CR in response to the prompt "UNIBUS CONTROLLER?". To specify a single test to be performed, type Y CR in response to "TYPE Y TO SELECT A TEST". Typing anything else will perform all tests, 1 to 20, on each drive that is ready (all tests will run on one drive before starting on another). If a single test was selected, the display will prompt "ENTER TEST NUMBER OR REQUEST MENU". Type M CR to have a list of tests displayed or type the number followed by a carriage return to specify a test. After testing has started, a CTRL C will cause "TYPE R TO RESTART THE DIAGNOSTIC OR E TO LIST THE ERROR LOG. ANYTHING ELSE WILL CONTINUE THE SAME TEST" to be displayed.

NOTE

CR designates a carriage return.

Possible responses are:

R CR -The diagnostic is restarted

E CR -The error log is displayed with errors listed by track number and error code number. To freeze the error log display, type CTRL S and to resume it type CTRL Q.

CR -The same test is started

NOTE

A CTRL C during a format of set media density operation is not acknowledged until the operation is completed.

2.0 Before a test is executed, the number of the test is displayed. A brief description of each test is given below. Test numbers are in octal.

2.1 Test 1

The test checks register communication.

1. 137776 is output to RXCS and the expected values of 5560 in RXCS and 33766 in RXDB are checked.

2. 177777 is output to RXDB and the expected value of 173767 is checked.

3. 0 is output to RXDB and the expected value of 0 is checked.

2.2 Test 2

The test checks that the initialize function sets RXCS correctly.

The initialize bit (bit 14) of RXCS is set and the initialize done and drive ready bits of RXCS are checked to verify that they are set (4040 in RXCS).

2.3 Test 3

The test checks controller interrupts.

RXCS is cleared, then interrupt enable (bit 6) is set. The number of interrupts is then checked to verify that exactly one interrupt occurs.

2.4 Test 4

The test checks the fill and empty buffer functions.

Various patterns are used to perform the fill buffer. Empty buffer is then operated and the data compared with the generating pattern.

The patterns used for the test are:

177777	052525	000000
125252	033333	

2.5 Test 5

The test checks CRC generation.

CRC generation is checked using the following data patterns for sector data:

Pattern	CRC Value
000000	024510
125252	163776
052525	047023
033333	137265

2.6 Test 6

The test checks head postioning.

The heads are stepped one track at a time from track Ø to track 76 and back again. A read sector is performed on each track to confirm that the heads are positioned on the correct track. Both sides are tested on double-sided equipment.

2.7 Test 7

The test checks single density disk formatting.

The diskette is formatted in single density and then checked for bad blocks. Both sides are tested on double-sided equipment.

2.10 Test 10

The test checks double density disk formatting.

The diskette is formatted in double density and then checked for bad blocks. Both sides are tested on double-sided equipment.

2.11 Test 11

The test checks illegal track and sector processing.

2.11.1

A read command is issued for track 77. RXCS is checked to ensure that the error and done bits are set (bits 15 and 5, respectively). RXDB is checked to ensure that no status bits are incorrectly set. A read error code is performed to verify that the error code is 40.

2.11.2

A read command is issued for sector Ø. RXCS is checked to ensure that the error and done bits are set, and RXDB is checked to ensure that no bits are set incorrectly. A read error code is performed to verify that the error code is 70.

2.12 Test 12

The test writes to random sectors and verifies the data.

Various patterns are used to write random sectors of random tracks (and random side for double-sided equipment). After each write the same sector is read to determine that the data are correct.

2.13 Test 13

The test checks deleted data writes.

A fill buffer is done, and a write sector with deleted data is performed on a random sector and track (and a random side on double-sided equipment). The data are then read from the sector and the deleted data bit (bit 6) of RXDB is checked to ensure that it is set. Finally, an empty buffer is performed to ensure the sector data are correct.

2.14 Test 14

The test checks that overwriting in a different density will not cause errors.

2.14.1

The diskette is formatted in double density. A set media density is done in single density and a bad block check is performed to ensure that no errors occurred.

2.14.2

The diskette is formatted in single density. A set media density is done in double density and a bad block check is performed to ensure that no errors occurred.

2.15 Test 15

The test checks that RAM addressing operates correctly.

2.15.1

A double density fill buffer is performed with all zeroes. A single density fill buffer with all ones is done, and then RAM is checked to ensure only the single density buffer region has changed.

2.15.2

A double density fill buffer is performed with all ones. A single density fill buffer with all zeroes is done, and then RAM is checked to ensure only the single density buffer region has changed.

2.16 Test 16

The test checks that a predetermined pattern can be written to random sectors.

The buffer data consist of each word containing its offset. The buffer data are then written to random sectors of random tracks (and random sides on double-sided equipment). The data are then read back to verify that the sector write was performed correctly.

NOTE

In particular, this test will verify that the special case of exactly four consecutive data one bits in double density is processed correctly, See paragraph 1.3.6 of the MXV21 Disk Controller Manual.

2.17 Test 17

The test checks that sector 1, track 1 of drive Ø is correctly read during initialization.

The test is only performed if a diskette is inserted in drive Ø. Sector 1 of track 1 is written with all ones; an initalize is then performed, followed by an empty buffer. The data are then checked to ensure they consist of all ones.

2.20 Test 20

The test checks non-existent memory processing.

The test performs an empty buffer with a bus location that ensures that non-existent memory is accessed. The error bit (bit 15) of RXCS and the NXM bit (bit 8) of RXDB are checked to make sure they are set.

2.21 Test 21

The test verifies that random sectors can be written and read successfully.

A random value is written in each word of the buffer. The data are written on random sectors of random tracks (and random sides on double-sided equipment). The data are then read and compared to the original data to verify that the operation was performed correctly.

2.22 Test 22

The test checks media wear.

The test loads the heads at tracks 0, 76, 1, 75, etc. The test attempts to load the heads at the same spot on each track. After the 77 head loads a bad block check is performed to check for media wear.

2.23 Test 23

The test aids in head alignment testing.

The test will position the heads to a selected track to assist in head alignment testing.

3.0 The following messages are displayed to explain detected errors:

3.1 RXCS BITS IN ERROR

```
   PC          EXP STA        REC STA
 dddddd         dddddd         dddddd
```

Where PC=program counter
EXP STA=expected RXCS
REC STA=received RXCS

3.2 RXDB BITS IN ERROR

```
   PC          EXP STA        REC STA
 dddddd        dddddd         dddddd
```

Where PC=program counter
EXP STA=expected RXDB
REC STA=received RXDB

3.3 BAD CRC CALCULATION

```
   PC          VALID          RCVD
 dddddd        dddddd         dddddd
```

Where PC=program counter
VALID=expected CRC value
RCVD=calculated CRC value

3.4 MULTIPLE INTERRUPTS

```
   PC          VALID          RCVD
 dddddd          1            dddddd
```

Where PC=program counter
VALID=number of expected interrupts=1
RCVD=number of interrupts that occurred

3.5 NO INTERRUPTS

LOCATION
dddddd

Where LOCATION=program counter

3.6 DELETED DATA BIT NOT SET

PC	STATUS
dddddd	dddddd

Where PC=program counter
STATUS=received RXDB

3.7 ILLEGAL SECTOR NOT DETECTED

PC	EXP STA	REC STA
dddddd	100040	dddddd

Where PC=program counter
EXP STA=expected RXCS (don't care bits masked off)
REC STA=received RXCS (don't care bits masked off)

3.8 WRONG ERROR CODE

PC	VALID	RCVD
dddddd	dddddd	dddddd

Where PC=program counter
VALID=expected error code
RCVD=received error code

3.9 INIT DONE OR DRIVE READY NOT SET

PC	EXP STA	REC STA
dddddd	204	dddddd

Where PC=program counter
EXP STA=expected RXDB (don't care bits masked off)
REC STA=received RXDB (don't care masked off)

3.10 DATA FROM FILL/EMPTY BUFFER DON'T MATCH

```
  PC      GD LOC     G DATA     BD LOC     B DATA
dddddd    dddddd     dddddd     dddddd     dddddd
```

Where PC=program counter
GD LOC=address of value used for fill buffer
G DATA=value used for fill buffer
BD LOC=address of non-comparing value
B DATA=non-comparing value

3.11 SEEK ERROR

```
  PC      REQTRK     CURTRK
dddddd    dddddd     dddddd
```

Where PC=program counter
REQTRK=requested track
CURTRK=track where heads stopped

3.12 RAM ADDRESSING ERROR

```
  PC      GD LOC     G DATA     BD LOC     B DATA
dddddd    dddddd     dddddd     dddddd     dddddd
```

Where PC=program counter
GD LOC=address of value used for fill buffer.
G DATA= value used for fill buffer
BD LOC= address of non-comparing value
B DATA= non-comparing value

3.13 CAN'T READ SECTOR HEADER ON TRACK

```
  PC      TRACK
dddddd     dd
```

Where PC=program counter
TRACK=track that contains bad sector header

3.14 ERROR DURING READ SECTOR

```
  PC      STATUS     CODE     SIDE     TRACK     SECTOR
dddddd    dddddd     ddd       d        dd        dd
```

Where PC=program counter
STATUS=received RXCS
CODE=error code
and SIDE, TRACK, SECTOR identify the sector that couldn't be read

3.15 ERROR DURING WRITE SECTOR

```
  PC      STATUS     CODE     SIDE     TRACK     SECTOR
dddddd    dddddd     ddd       d        dd        dd
```

Where PC=program counter
STATUS=received RXCS
CODE=error code
and SIDE, TRACK, SECTOR identify the sector where the write sector operation failed.

3.16 NXM BIT NOT SET

```
  PC      EXP STA    REC STA
dddddd    dddddd     dddddd
```

Where PC=program counter
EXP STA=expected RXCS
REC STA=received RXCS

3.17 ERROR BIT NOT SET

```
  PC      EXP STA    REC STA
dddddd    dddddd     dddddd
```

Where PC=program counter
EXP STA=expected RXCS
REC STA=received RXCS

3.18 TIMEOUT OR BUS ERROR

TRAP PC
dddddd

Where TRAP PC=location where a trap through location 4 occurred

3.19 ILLEGAL INSTRUCTION

TRAP PC
dddddd

Where TRAP PC=location where a trap through location 10 occurred

3.20 ILLEGAL INTERRUPT

VECTOR
dddddd

Where VECTOR=location of the vector of the illegal interrupt

DIAGNOSTIC LISTING

4.0

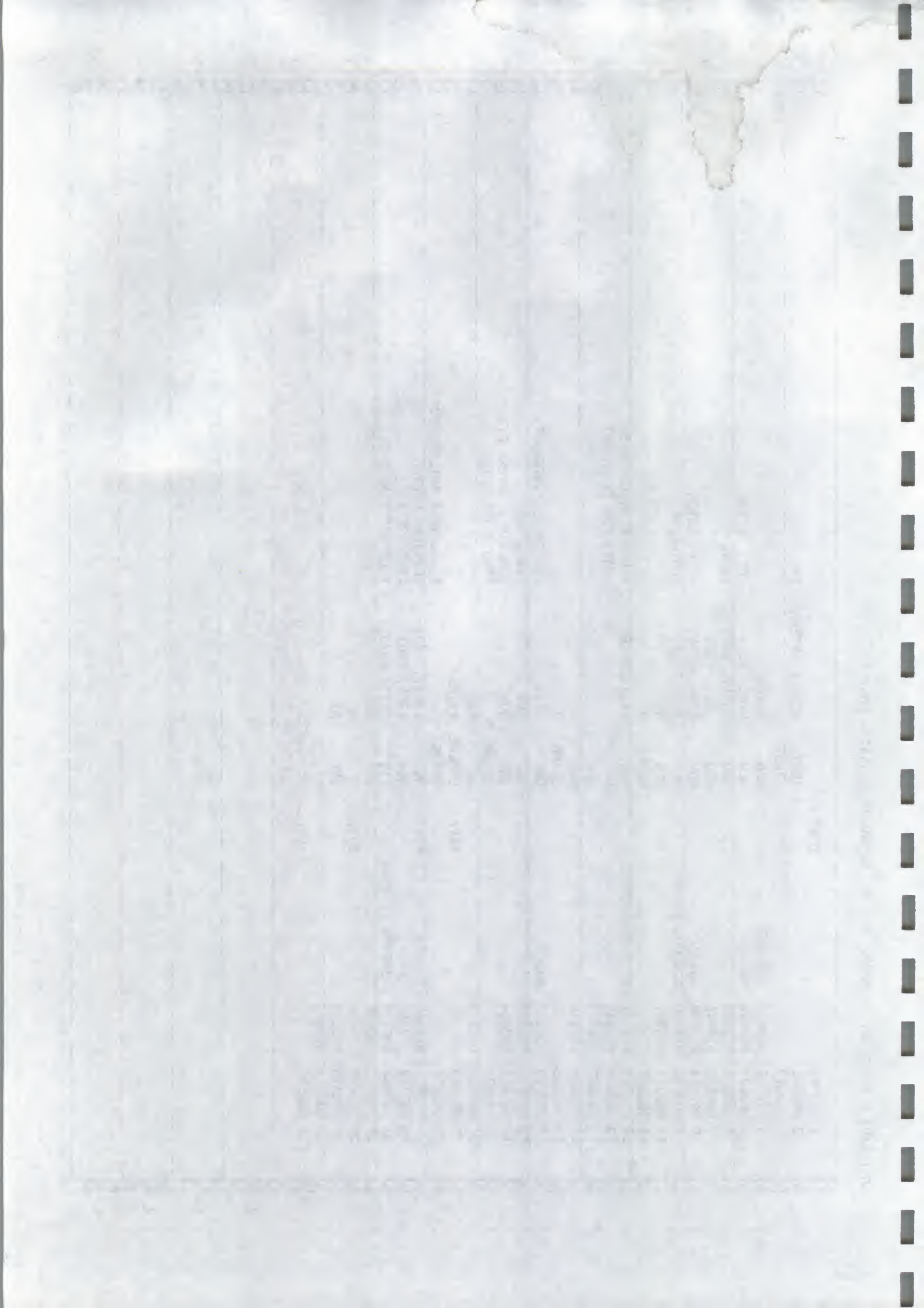
The following pages contain the listing of the Diagnostic. The program counter values that appear in the error messages correspond to locations in the listing.

```
                        .TITLE  MTI RX02 DIAGNOSTIC
                ; COPYRIGHT (C) 1983
                ;
                ; MICRO TECHNOLOGY INCORPORATED (MTI)
                ; 1620 MIRALOMA AVE
                ; PLACENTIA, CALIFORNIA  92670
                ;
                ; THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY
                ; ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH
                ; THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS
                ; SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED
                ; OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT
                ; FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE
                ; LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE
                ; SHALL AT ALL TIMES REMAIN IN MTI.
                ;
                ; THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
                ; AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY
                ; MICRO TECHNOLOGY INCORPORATED. MTI ASSUMES NO RESPONSIBILTY
                ; FOR ANY ERRORS THAT MAY OCCUR IN THIS SOFTWARE.
                ;
                ; MTI ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY
                ; OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY MTI.
```

```
                                        .ENABL  AMA
                                        .NLIST  ME
                                        .NLIST  CND
002000  000137  000000G                 JMP     START#          ;ENTRY POINT AT 2000
002004                          TEST1:
002004                                  TEST    <CHECK RXCS AND RXDB>
002004                                  STEST
002012  013700  000000G                 MOV     RXCS,R0         ;ADDRESS OF STATUS REGISTER
002016  012710  137776                  MOV     #137776,(R0)
002022  000240                          NOP                     ;DELAY
002024  000240                          NOP
002026  012001                          MOV     (R0)+,R1        ;SAVE FOR ERROR MESSAGE
002030  020137  000000G                 CMP     R1,CCSBIT       ;PROPER BITS SET?
002034  001413                          BEQ     10$             ;BR IF SO
002036                                  SVALUE  CCSBIT,R1       ;INSERT VALUES IN ERROR MESSAGE
002056                                  ERRMSG  1               ;'RXCS BITS IN ERROR'
002064                          10$:
002064  011001                          MOV     (R0),R1
002066  020137  000000G                 CMP     R1,CDBBIT       ;PROPER BITS SET?
002072  001413                          BEQ     20$             ;BR IF SO
002074                                  SVALUE  CDBBIT,R1
002114                                  ERRMSG  2               ;'RXDB BITS IN ERROR'
002122                          20$:
002122  012710  177777                  MOV     #-1,(R0)
002126  000240                          NOP
002130  000240                          NOP
002132  011001                          MOV     (R0),R1
002134  020137  000000G                 CMP     R1,DDBBIT
002140  001413                          BEQ     30$
002142                                  SVALUE  DDBBIT,R1
002162                                  ERRMSG  2
002170                          30$:
002170  005010                          CLR     (R0)
002172  000240                          NOP
002174  000240                          NOP
002176  011001                          MOV     (R0),R1
002200  001413                          BEQ     40$
002202                                  SVALUE  #0,R1
002222                                  ERRMSG  2
002230                          40$:
```

```
                                ; THIS TEST CHECKS THAT INITIALIZE DONE AND
                                ; DRIVE READY ARE CORRECTLY SET AFTER PERFORMING
                                ; AN INITIALIZE COMMAND.
002230                          TEST2:
002230                                  TEST    <CHECK INITIALIZE COMMAND>
002234                                  STEST
002242  012777  000006 000006           MOV     #BIT14,@RXCS    ;INITIALIZE
002250  004737  000006                  JSR     PC,CKINIT       ;CHECK INITIALIZE DONE, DRIVE READY BITS
002254  103014                          BCC     99$             ;BR IF BITS OKAY
002256                                  SVALUE  #204,ESTAT
002300                                  ERRMSG 15               ;'INIT DONE/DRIVE READY NOT SET'
002306                          99$:
```

```
                                TEST3:
                                        TEST    <CHECK INTERRUPTS>
                                        STEST
                                        CLR     @RXCS
                                        MOV     RXVEC,R0        ;GET VECTOR
                                        MOV     (R0)+,-(SP)     ;SAVE
                                        MOV     (R0),-(SP)
                                        MOV     #50$,-(R0)      ;SET VECTOR
                                        MOV     #PR7,2(R0)      ;PRIORITY 7
                                        CLR     R1
                                        SETPR   0
                                        MOV     #BIT6,@RXCS     ;ENABLE INTERRUPTS
                                        NOP                     ;WAIT FOR INTERRUPT
                                        NOP
                                        NOP
                                        SETPR   7
                                        CMP     R1,#1           ;CHECK FOR INTERRUPT
                                        BEQ     40$             ;BR IF OK
                                        BGT     30$             ;MULTIPLE INTERRUPTS
                                        ERRMSG 5                ;'NO INTERRUPTS'
                                        BR      40$
                                30$:    SVALUE  #1,R1
                                        ERRMSG 4                ;'MULTIPLE INTERRUPTS'
                                40$:    MOV     (SP)+,2(R0)     ;RESTORE VECTOR
                                        MOV     (SP)+,(R0)      ;RESTORE PSW
                                        BIC     #BIT6,@RXCS     ;CLEAR INTERRUPTS
                                        BR      60$
                                50$:
                                        INC     R1
                                        RTI
                                60$:
```



```
                        ; THIS TEST WILL CHECK FILL AND EMPTY BUFFER FOR DIFFERENT
                        ; PATTERNS.
002500                  TEST4:
002500                          TEST    <TEST FILL/EMPTY BUFFER FOR DIFFERENT PATTERNS>
002504                          STEST
002512  012701  000000G         MOV     #PATRN1,R1      ;TABLE OF PATTERNS
002516                  10$:
002516  012104                  MOV     (R1)+,R4
002520  020104                  CMP     R1,R4           ;DONE?
002522  001422                  BEQ     99$             ;BR IF SO
002524  012703  000000G         MOV     #BUF1,R3
002530  004737  000000G         JSR     PC,BUFFIL       ;FILL THE BUFFER WITH THE PATTERN
002534  013746  000000G         MOV     DENS,-(SP)      ;SAVE THE DENSITY
002540  012737  000400  000000G MOV     #400,DENS       ;SET DOUBLE DENSITY
002546  004737  000000G         JSR     PC,FILLBF       ;FILL BUFFER
002552  004737  000000G         JSR     PC,EMPBF        ;EMPTY BUFFER
002556  012637  000000G         MOV     (SP)+,DENS      ;RESTORE DENSITY
002562  004737  000000G         JSR     PC,CMPBF        ;COMPARE FILL AND EMPTY DATA
002566  000753                  BR      10$
002570                  99$:
```

```
                                ; THIS TEST WILL TEST CRC GENERATION BY DOING A SINGLE
                                ; DENSITY FILL BUFFER AND A DOUBLE DENSITY EMPTY BUFFER
                                ; AND CHECKING THAT THE CRC IS VALID.
002570                          TEST5:
002570                                  TEST    <TEST CRC>
002574                                  STEST
002602  005037  000000G                 CLR     B22FLG          ;CLEAR 22 BIT ADDRESSING FLAG
002606  013700  000000G                 MOV     RXCS,R0         ;CSR ADDRESS
002612  012720  000001                  MOV     #1,(R0)+        ;FILL BUFFER
002616  004737  000000G                 JSR     PC,TRWAIT       ;WAIT FOR TRANSFER READY BIT
002622  032777  002000  000000G         BIT     #2000,@RXCS     ;22 BIT ADDRESSING?
002630  001402                          BEQ     2$              ;BR IF NOT
002632  005237  000000G                 INC     B22FLG
002636                          2$:
002636  005010                          CLR     (R0)            ;WORD COUNT 0, SO ZERO FILL
002640  004737  000000G                 JSR     PC,TRWAIT
002644  005010                          CLR     (R0)            ;LOCATION
002646  005737  000000G                 TST     B22FLG          ;22 BIT ADDRESSING?
002652  001403                          BEQ     4$              ;BR IF NOT
002654  004737  000000G                 JSR     PC,TRWAIT       ;WAIT FOR TRANSFER READY
002660  005010                          CLR     (R0)
002662                          4$:
002662  004737  000000G                 JSR     PC,DNWAIT       ;WAIT FOR DONE
002666  103012                          BCC     10$             ;CONTROLLER READY
002670                          5$:
002670  012737  000001  000000G         MOV     #1,REG0
002676  017737  000000G 000000G         MOV     @RXCS,REG2
002704                                  ERRMSG  10
002712  000512                          BR      99$
002714                          10$:
002714  012760  000403  177776         MOV     #403,-2(R0)     ;DOUBLE DENSITY EMPTY
002722  004737  000000G                 JSR     PC,TRWAIT       ;WAIT FOR TR
002726  012710  000200                  MOV     #200,(R0)       ;WORD COUNT
002732  004737  000000G                 JSR     PC,TRWAIT
002736  012710  000000G                 MOV     #BUF1,(R0)      ;LOCATION
002742  005737  000000G                 TST     B22FLG          ;22 BIT ADDRESSING?
002746  001403                          BEQ     15$             ;BR IF NOT
002750  004737  000000G                 JSR     PC,TRWAIT       ;WAIT FOR TRANSFER READY
002754  005010                          CLR     (R0)
002756                          15$:
002756  004737  000000G                 JSR     PC,DNWAIT
002762  103742                          BCS     5$              ;TIMED-OUT
```

```
    43 002764  023727  000200G 024510          CMP     BUF1+200,#24510 ;CRC OKAY?
    44 002772  001415                          BEQ     20$             ;BR IF SO
    45 002774                                  SVALUE  #24510,BUF1+200
    46 003016                                  ERRMSG  3               ;'INVALID CRC'
    47 003024  000445                          BR      99$
    48 003026                          20$:
    49 003026  012701  000000G                 MOV     #PATRN,R1       ;TABLE OF PATTERNS FOR CRC TEST
    50 003032  012700  000000G                 MOV     #CRCVAL,R0      ;TABLE OF VALID CRC'S
    51 003036                          30$:
    52 003036  012104                          MOV     (R1)+,R4
    53 003040  001437                          BEQ     99$             ;BR IF DONE
    54 003042                          40$:
    55 003042  012703  000000G                 MOV     #BUF1,R3
    56 003046  004737  000000G                 JSR     PC,BUFFIL       ;FILL THE BUFFER WITH THE PATTERN
    57 003052  013746  000000G                 MOV     DENS,-(SP)      ;SAVE THE DENSITY
    58 003056  005037  000000G                 CLR     DENS            ;SET SINGLE DENSITY
    59 003062  004737  000000G                 JSR     PC,FILLBF       ;FILL BUFFER
    60 003066  012737  000400  000000G         MOV     #400,DENS       ;SET DOUBLE DENSITY
    61 003074  004737  000000G                 JSR     PC,EMPBF        ;EMPTY BUFFER
    62 003100  012637  000000G                 MOV     (SP)+,DENS      ;RESTORE DENSITY
    63 003104  023720  000200G                 CMP     BUF2+200,(R0)+  ;CRC VALID?
    64 003110  001752                          BEQ     30$             ;BR IF SO
    65 003112                                  SVALUE  -(R0),BUF2+200
    66 003132                                  ERRMSG  3
    67 003140                          99$:
```

```
                                        ; THIS TEST WILL STEP FROM TRACK 0 TO TRACK 76 AND BACK AGAIN
003140                                  TEST6:
003140                                          TEST    <TRACK POSITIONING TEST>
003144                                          STEST
003152  005037  000000G                         CLR     SIDE2
003156                                  5$:
003156  005037  000000G                         CLR     DIR
003162  105037  000000G                         CLRB    TRK
003166  112737  000001  000000G                 MOVB    #1,SECT
003174                                  10$:
003174  005237  000000G                         INC     IGNERR          ;INHIBIT ERROR MESSAGE IN READ SECTOR ROUTINE
003200  004737  000000G                         JSR     PC,READ         ;READ TO POSITION HEADS TO TRACK
003204  103016                                  BCC     15$             ;BR IF NO ERROR
003206  005046                                  CLR     -(SP)
003210  113716  000000G                         MOVB    TRK,(SP)
003214                                          SVALUE  (SP)+
003226                                          ERRMSG  23              ;'CAN'T READ SECTOR HEADER ON TRACK'
003234  004737  000000G                         JSR     PC,IOERR        ;ACCUMULATE ERROR TOTALS
003240  000404                                  BR      20$
003242                                  15$:
003242  004737  000000G                         JSR     PC,READEC       ;READ ERROR CODE TO GET CURRENT TRACK
003246  004737  000000G                         JSR     PC,CKTRK        ;CHECK THAT TRACK MATCHES SELECTED TRACK
003252                                  20$:
003252  005037  000000G                         CLR     IGNERR
003256  005737  000000G                         TST     DIR             ;STEPPING IN?
003262  001013                                  BNE     50$             ;BR IF NOT
003264  105237  000000G                         INCB    TRK
003270  123727  000000G 000115                 CMPB    TRK,#77.        ;DONE?
003276  002736                                  BLT     10$             ;BR IF NOT
003300  005237  000000G                         INC     DIR
003304  105337  000000G                         DECB    TRK
003310  000731                                  BR      10$
003312                                  50$:
003312  105337  000000G                         DECB    TRK             ;DONE?
003316  100326                                  BPL     10$             ;BR IF NOT
003320  005737  000000G                         TST     SIDES           ;SINGLE SIDE?
003324  001406                                  BEQ     99$             ;BR IF SO
003326  005737  000000G                         TST     SIDE2           ;FINISHED SECOND SIDE?
003332  001003                                  BNE     99$             ;BR IF SO
003334  005237  000000G                         INC     SIDE2
003340  000706                                  BR      5$
003342                                  99$:
```

```
                                        ; THIS TEST WILL FORMAT THE DISKETTE TO SINGLE DENSITY
                                        ;AND CHECK FOR BAD BLOCKS
003342                                  TEST7:
003342                                          TEST    <FORMAT SINGLE DENSITY AND CHECK BLOCKS>
003346                                          STEST
003354  042737  000400  000000G               BIC     #400,DENS       ;SINGLE DENSITY
003362  004737  000000G                       JSR     PC,FRMT         ;FORMAT THE DISKETTE
003366  004737  000000G                       JSR     PC,BLKCHK       ;CHECK FOR BAD BLOCKS


                                        ; THIS TEST WILL FORMAT THE DISKETTE TO DOUBLE DENSITY
                                        ;AND CHECK FOR BAD BLOCKS
003372                                  TEST10:
003372                                          TEST    <FORMAT DOUBLE DENSITY AND CHECK BLOCKS>
003376                                          STEST
003404  052737  000400  000000G               BIS     #400,DENS       ;DOUBLE DENSITY
003412  004737  000000G                       JSR     PC,FRMT
003416  004737  000000G                       JSR     PC,BLKCHK
```

```
                                ; THIS TEST WILL CHECK THAT AN ILLEGAL TRACK OR SECTOR
                                ;WILL BE PROCESSED AS AN ERROR
003422                          TEST11:
003422                                  TEST    <TEST ILLEGAL TRACK AND SECTOR>
003426                                  STEST
003434  005037  000000G                CLR     SIDE2
003440  005237  000000G                INC     IGNERR          ;INHIBIT ERROR MESSAGE IN READ SECTOR ROUTINE
003444  112737  000001  000000G        MOVB    #1,SECT         ;SECTOR 1
003452  112737  000115  000000G        MOVB    #77.,TRK        ;TRACK 77
003460  004737  000000G                JSR     PC,READ         ;TRY TO READ
003464  013703  000000G                MOV     STATUS,R3       ;LOAD STATUS
003470  042703  037537                BIC     #37537,R3
003474  022703  100040                CMP     #100040,R3      ;CK STATUS RETURNED
003500  001413                        BEQ     4$              ;BR IF CORRECT
003502                                  SVALUE  #100040,R3
003522                                  ERRMSG 1                ;'STATUS ERROR'
003530                          4$:
003530  032737  000105  000000G        BIT     #105,ESTAT      ;CHECK ERROR STATUS
003536  001417                        BEQ     8$              ;BR IF CORRECT
003540  013746  000000G                MOV     ESTAT,-(SP)
003544  042716  177672                BIC     #177672,(SP)    ;MASK OFF UNWANTED BITS
003550                                  SVALUE  #0,(SP)+
003570                                  ERRMSG 1                ;'STATUS ERROR'
003576  013703  000000G        8$:     MOV     ECTBPT,R3
003602  022713  000040                CMP     #40,(R3)        ;ERROR CODE 40 FOR ILLEGAL TRACK
003606  001415                        BEQ     12$             ;BR IF CORRECT
003610  005002                        CLR     R2
003612  111302                        MOVB    (R3),R2         ;DISPLAY INCORRECT ERROR CODE
003614                                  SVALUE  #40,R2
003634                                  ERRMSG 14               ;WRONG ERROR CODE
003642                          12$:
003642  105037  000000G                CLRB    SECT            ;SECTOR 0
003646  105037  000000G                CLRB    TRK             ;TRACK 0
003652  004737  000000G                JSR     PC,READ
003656  013703  000000G                MOV     STATUS,R3       ;LOAD STATUS
003662  042703  037537                BIC     # 37537,R3
003666  022703  100040                CMP     #100040,R3      ;CHECK FOR ERROR AND DONE
003672  001413                        BEQ     14$
003674                                  SVALUE  #100040,R3
003714                                  ERRMSG 13               ;'ILLEGAL SECTOR NOT DETECTED'
003722  013702  000000G        14$:    MOV     ESTAT,R2        ;GET ERROR
003726  042702  140742                BIC     #140742,R2      ;CLEAR INTERRUPT PENDING
```

```
    43 003732  005702                          TST     R2              ;STATUS OK
    44 003734  001413                          BEQ     18$             ;BR IF SO
    45 003736                                  SVALUE  #0,R2
    46 003756                                  ERRMSG  1               ;"STATUS ERROR"
    47 003764  013703  000000G          18$:   MOV     ECTBPT,R3
    48 003770  022713  000070                  CMP     #70,(R3)        ;ERROR CODE 70 FOR ILLEGAL TRACK
    49 003774  001415                          BEQ     22$             ;BR IF OK
    50 003776  005002                          CLR     R2
    51 004000  111302                          MOVB    (R3),R2         ;DISPLAY INCORRECT ERROR CODE
    52 004002                                  SVALUE  #70,R2
    53 004022                                  ERRMSG  14              ;ERROR CODE WRONG
    54 004030                          22$:
    55 004030  005037  000000G                 CLR     IGNERR
```

```
                                        ; THIS TEST WILL WRITE VARIOUS PATTERNS TO RANDOM SECTORS,
                                        ;READ THE DATA BACK AND COMPARE.
004034                                  TEST12:
004034                                          TEST    <TEST WRITE SECTOR USING RANDOM TRACK AND SECTORS>
004040                                          STEST
004046  012746  000144                          MOV     #100.,-(SP)     ;WRITE 100 SECTORS
004052                                  5$:
004052  012702  000000G                         MOV     #PATRN2,R2      ;TABLE OF PATTERNS
004056                                  10$:
004056  005716                                  TST     (SP)            ;DONE?
004060  001002                                  BNE     15$             ;BR IF NOT
004062  000137  004154'                         JMP     99$
004066                                  15$:
004066  012204                                  MOV     (R2)+,R4
004070  020204                                  CMP     R2,R4           ;DONE WITH PATTERNS?
004072  001767                                  BEQ     5$              ;START OVER IF SO
004074  005316                                  DEC     (SP)
004076  012703  000000G                         MOV     #BUF1,R3
004102  004737  000000G                         JSR     PC,BUFFIL       ;FILL THE BUFFER WITH THE PATTERN
004106  004737  000000G                         JSR     PC,FILLBF       ;FILL THE BUFFER
004112  004737  000000G                         JSR     PC,RANTRK       ;GET A RANDOM TRACK
004116  004737  000000G                         JSR     PC,RANSEC       ;GET A RANDOM SECTOR
004122  004737  000000G                         JSR     PC,RANSID       ;GET A RANDOM SIDE IF DOUBLE SIDED
004126  004737  000000G                         JSR     PC,WRITE
004132  103751                                  BCS     10$
004134  004737  000000G                         JSR     PC,READ
004140  103746                                  BCS     10$
004142  004737  000000G                         JSR     PC,EMPBF        ;EMPTY BUFFER
004146  004737  000000G                         JSR     PC,CMPBF        ;COMPARE FILL AND EMPTY DATA
004152  000741                                  BR      10$
004154                                  99$:
004154  005726                                  TST     (SP)+           ;RESTORE STACK
```

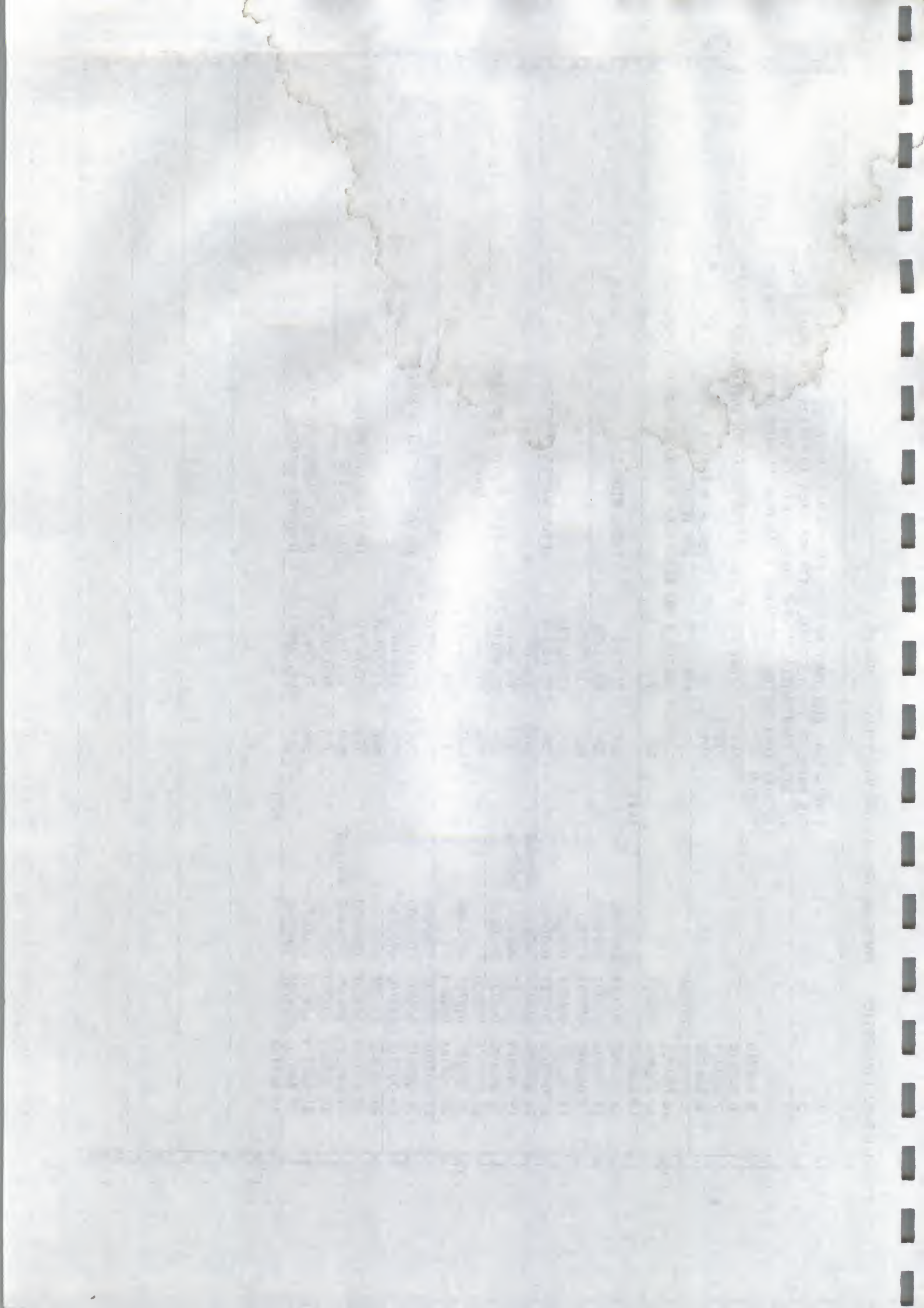
```
                        ; THIS TEST WILL WRITE A DELETED DATA SECTOR AND THEN
                        ;READ THE SECTOR TO ENSURE THAT THE DELETED DATA BIT IS
                        ;SET AND THE BUFFER CONTENTS ARE VALID.
004156                  TEST13:
004156                          TEST    <DELETED DATA TEST>
004162                          STEST
004170  012746  000012          MOV     #10.,-(SP)      ;NUMBER OF ATTEMPTS
004174                  5$:
004174  004737  000000G         JSR     PC,FILLBF       ;FILL BUFFER
004200  004737  000000G         JSR     PC,RANTRK
004204  004737  000000G         JSR     PC,RANSEC
004210  004737  000000G         JSR     PC,RANSID
004214  004737  000000G         JSR     PC,WRITDD       ;WRITE DELETED DATA
004220  103412                  BCS     10$
004222  004737  000000G         JSR     PC,READ
004226  103407                  BCS     10$
004230  004737  000000G         JSR     PC,CKDD         ;CHECK THE DELETED DATA BIT
004234  004737  000000G         JSR     PC,EMPBF        ;EMPTY BUFFER
004240  004737  000000G         JSR     PC,CMPBFA       ;CHECK DATA
004244  000402                  BR      99$             ;EXIT
004246                  10$:
004246  005316                  DEC     (SP)            ;TRY AGAIN?
004250  001351                  BNE     5$              ;BR IF SO
004252                  99$:
004252  005726                  TST     (SP)+           ;RESTORE STACK POINTER
```

```
                                        ; THIS TEST WILL:
                                        ; 1. FORMAT DOUBLE DENSITY, SET MEDIA SINGLE DENSITY ,BLOCK CHECK
                                        ; 2. FORMAT SINGLE DENSITY, SET MEDIA DOBULE DENSITY, BLOCK CHECK
004254                                  TEST14:
004254                                          TEST    <FORMAT, SET MEDIA DENSITY, BLOCK CHECK>
004260                                  STEST
004266  052737  000400  000000G        BIS     #400,DENS       ;DOUBLE DENSITY
004274  004737  000000G                JSR     PC,FRMT         ;FORMAT
004300  042737  000400  000000G        BIC     #400,DENS       ;SINGLE DENSITY
004306  005237  000000G                INC     SMEDIA
004312  004737  000000G                JSR     PC,FRMT         ;SET MEDIA DENSTIY
004316  005037  000000G                CLR     SMEDIA
004322  004737  000000G                JSR     PC,BLKCHK       ;BLOCK CHECK
004326  004737  000000G                JSR     PC,FRMT
004332  052737  000400  000000G        BIS     #400,DENS
004340  005237  000000G                INC     SMEDIA
004344  004737  000000G                JSR     PC,FRMT
004350  005037  000000G                CLR     SMEDIA
004354  004737  000000G                JSR     PC,BLKCHK
```

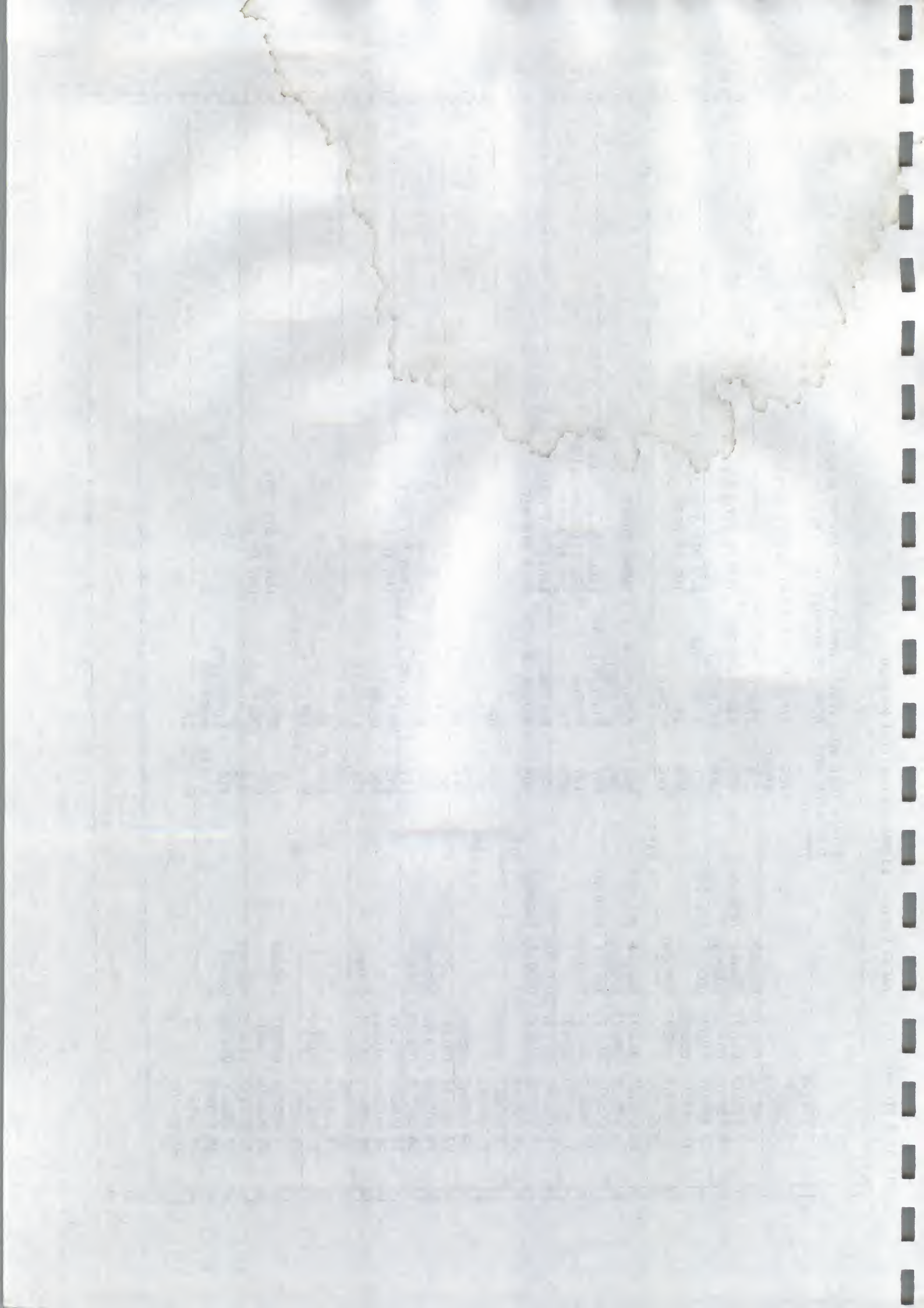
```
                                ; THIS TEST WILL DETERMINE IF FAULTY RAM ADDRESSING IS CAUSING
                                ;BITS TO BE SET IN HIGH AND LOW SECTOR BUFFER AREAS OF RAM.
004360                          TEST15:
004360                                  TEST    <RAM ADDRESSING TEST>
004364                                  STEST
004372  005237  000000G                INC     IGNERR
004376  005004                  CLR     R4              ;PATTERN
004400  012703  000000G          MOV     #BUF1,R3
004404  004737  000000G          JSR     PC,BUFFIL       ;FILL BUFFER WITH ZEROES
004410                          PUSH    DENS            ;SAVE DENSITY
004414  052737  000400  000000G  BIS     #400,DENS       ;DOUBLE DENSITY
004422  004737  000000G          JSR     PC,FILLBF       ;DOUBLE DENSITY FILL WITH ZEROES
004426  012704  177777          MOV     #177777,R4
004432  012703  000000G          MOV     #BUF1,R3
004436  004737  000000G          JSR     PC,BUFFIS       ;FILL FIRST HALF WITH ONES
004442  042737  000400  000000G  BIC     #400,DENS
004450  004737  000000G          JSR     PC,FILLBF       ;SINGLE DENSITY FILL WITH ONES
004454  052737  000400  000000G  BIS     #400,DENS
004462  004737  000000G          JSR     PC,EMPBF        ;DOUBLE DENSITY EMPTY
004466  005037  000200G          CLR     BUF2+200        ;CRC WORD
004472  004737  000000G          JSR     PC,CMPBFD
004476  012704  177777          MOV     #177777,R4
004502  012703  000000G          MOV     #BUF1,R3
004506  004737  000000G          JSR     PC,BUFFIL       ;FILL BUFFER WITH ONES
004512  052737  000400  000000G  BIS     #400,DENS       ;DOUBLE DENSITY
004520  004737  000000G          JSR     PC,FILLBF       ;DOUBLE DENSITY FILL WITH ONES
004524  005004                  CLR     R4
004526  012703  000000G          MOV     #BUF1,R3
004532  004737  000000G          JSR     PC,BUFFIS       ;FILL FIRST HALF WITH ZEROES
004536  042737  000400  000000G  BIC     #400,DENS
004544  004737  000000G          JSR     PC,FILLBF       ;SINGLE DENSITY FILL WITH ZEROES
004550  052737  000400  000000G  BIS     #400,DENS
004556  004737  000000G          JSR     PC,EMPBF        ;DOUBLE DENSITY EMPTY
004562  012737  177777  000200G  MOV     #177777,BUF2+200
004570  004737  000000G          JSR     PC,CMPBFD
004574  005037  000000G          CLR     IGNERR
004600                          POP     DENS            ;RESTORE DENSITY
```

```
                        ; THIS TEST WILL FILL EACH WORD IN THE BUFFER WITH
                        ;ITS OFFSET AND WRITE AND VERIFY THE DATA TO RANDOM
                        ;SECTORS OF THE DISK.
004604                  TEST16:
004604                          TEST    <WRITE DATA CONSISTING OF OFFSET NUMBER>
004610                          STEST
004616  012746  000144          MOV     #100.,-(SP)     ;NUMBER OF PASSES
004622  012703  000000G         MOV     #BUF1,R3        ;BUFFER LOC
004626  005004                  CLR     R4              ;PATTERN
004630                  10$:
004630  010423                  MOV     R4,(R3)+
004632  005724                  TST     (R4)+           ;INCREMENT PATTERN
004634  020427  000400          CMP     R4,#400         ;DONE WITH PATTERN?
004640  002773                  BLT     10$             ;BR IF NOT
004642                  20$:
004642  004737  000000G         JSR     PC,FILLBF       ;FILL BUFFER FUNCTION
004646  004737  000000G         JSR     PC,RANTRK       ;RANDOM TRACK
004652  004737  000000G         JSR     PC,RANSEC       ;RANDOM SECTOR
004656  004737  000000G         JSR     PC,RANSID       ;RANDOM SIDE IF DOUBLE SIDED
004662  004737  000000G         JSR     PC,WRITE
004666  004737  000000G         JSR     PC,READ
004672  004737  000000G         JSR     PC,EMPBF
004676  004737  000000G         JSR     PC,CMPBF        ;COMPARE BUFFERS
004702  005316                  DEC     (SP)            ;DONE WITH TEST?
004704  001356                  BNE     20$             ;BR IF NOT
004706  005726                  TST     (SP)+           ;RESTORE STACK POINTER
```

```
                        ; THIS TEST WILL ONLY BE PERFORMED IF A DISKETTE IS INSERTED
                        ;IN DRIVE 0. THE TEST WILL WRITE ALL ONES TO SECTOR 1 OF TRACK 1.
                        ;IT WILL THEN INITIALIZE THE DRIVE AND DO AN EMPTY BUFFER TO
                        ;VERIFY THAT SECTOR 1 OF TRACK 1 WAS READ BECAUSE OF THE INITIALIZE.
004710                  TEST17:
004710                          TEST    <TEST CONTROLLER BUFFER AFTER AN INITIALIZE>
004714                          STEST
004722                          TSTDR0                  ;DRIVE 0 READY?
004734  103036                  BCC     10$             ;BR IF SO
004736                          COMMENT <DRIVE 0 NOT READY-CAN'T READ SECTOR 1, TRACK 1>
005030  000451                  BR      99$
005032                  10$:
005032  005037  000000G           CLR     SIDE2           ;READ SIDE 1
005036  012703  000000G           MOV     #BUF1,R3
005042  012704  177777           MOV     #177777,R4
005046  004737  000000G           JSR     PC,BUFFIL
005052  004737  000000G           JSR     PC,FILLBF       ;FILL BUFFER FUNCTION
005056  112737  000001  000000G   MOVB    #1,TRK
005064  112737  000001  000000G   MOVB    #1,SECT
005072  004737  000000G           JSR     PC,WRITE        ;WRITE SECTOR 1 OF TRACK 1
005076  103426                  BCS     99$             ;BR IF ERROR
005100  012703  000000G           MOV     #BUF1,R3
005104  005004                  CLR     R4
005106  004737  000000G           JSR     PC,BUFFIL
005112  004737  000000G           JSR     PC,FILLBF       ;CLEAR CONTROLLER BUFFER
005116  012703  000000G           MOV     #BUF1,R3
005122  012704  177777           MOV     #177777,R4
005126  004737  000000G           JSR     PC,BUFFIL       ;RESTORE CPU BUFFER
005132  012777  000000G 000000G   MOV     #BIT14,@RXCS    ;INITIALIZE
005140  004737  000000G           JSR     PC,CKINIT
005144  004737  000000G           JSR     PC,EMPBF        ;EMPTY BUFFER FUNCTION
005150  004737  000000G           JSR     PC,CMPBFA       ;COMPARE BUFFERS
005154                  99$:
```



```
                                        ; THIS TEST WILL VERIFY THAT A NON-EXISTENT MEMORY
                                        ;LOCATION WILL CAUSE THE NXM BIT TO BE SET
005154                                  TEST20:
005154                                          TEST    <TEST NON-EXISTENT MEMORY PROCESSING>
005160                                          STEST
005166  005037  000000G                        CLR     B22FLG
005172  012777  030403  000000G                MOV     #30403,@RXCS    ;EMPTY BUFFER (A16, A17 SET)
005200  004737  000000G                        JSR     PC,TRWAIT
005204  032777  002000  000000G                BIT     #2000,@RXCS     ;22 BIT ADDRESSING?
005212  001402                                BEQ     2$              ;BR IF NOT
005214  005237  000000G                        INC     B22FLG
005220                                  2$:
005220  012777  000200  000000G                MOV     #200,@RXDB      ;WORD COUNT
005226  004737  000000G                        JSR     PC,TRWAIT
005232  012777  177600  000000G                MOV     #177600,@RXDB   ;ILLEGAL BUS ADDRESS
005240  005737  000000G                        TST     B22FLG          ;22 BIT ADDRESSING?
005244  001405                                BEQ     5$              ;BR IF NOT
005246  004737  000000G                        JSR     PC,TRWAIT       ;WAIT FOR TRANSFER READY
005252  012777  000017  000000G                MOV     #17,@RXDB       ;SET A18-A21
005260                                  5$:
005260  005005                                CLR     R5
005262                                  10$:
005262  005305                                DEC     R5
005264  001376                                BNE     10$
005266  017705  000000G                        MOV     @RXDB,R5
005272  032705  004000                        BIT     #4000,R5        ;NXM BIT SET?
005276  001016                                BNE     20$             ;BR IF SO
005300  010504                                MOV     R5,R4
005302  052704  004000                        BIS     #4000,R4
005306                                          SVALUE  R5,R4
005324                                          ERRMSG  26
005332  000422                                BR      30$
005334                                  20$:
005334  005777  000000G                        TST     @RXCS           ;ERROR BIT SET?
005340  100417                                BMI     30$             ;BR IF SO
005342  017705  000000G                        MOV     @RXCS,R5
005346  052705  100000                        BIS     #100000,R5
005352                                          SVALUE  @RXCS,R5
005372                                          ERRMSG  27
005400                                  30$:
```



```
                                ; THIS TEST WILL WRITE A RANDOM PATTERN TO RANDOM SECTORS,
                                ;PERFORM AN INIT TO HOME THE DRIVE AND VERIFY THE DATA.
005400                          TEST21:
005400                                  TEST    <WRITE RANDOM SECTOR DATA>
005404                                  STEST
005412  012746  000144                    MOV     #100.,-(SP)     ;NUMBER OF SECTORS
005416                          10$:
005416  004737  000000G                   JSR     PC,RANDOM       ;GET RANDOM VALUE
005422  013704  000000G                   MOV     RANVAL,R4
005426  012703  000000G                   MOV     #BUF1,R3
005432  004737  000000G                   JSR     PC,BUFFIL       ;FILL THE LSI BUFFER
005436  004737  000000G                   JSR     PC,FILLBF       ;PERFORM A FILL BUFFER FUNCTION
005442  004737  000000G                   JSR     PC,RANTRK       ;RANDOM TRACK
005446  004737  000000G                   JSR     PC,RANSEC       ;RANDOM SECTOR
005452  004737  000000G                   JSR     PC,RANSID       ;RANDOM SIDE IF DOUBLE SIDED
005456  004737  000000G                   JSR     PC,WRITE
005462  103416                          BCS     40$
005464  012777  040000  000000G          MOV     #40000,@RXCS    ;INITIALIZE FUNCTION
005472                          30$:
005472  032777  000000G 000000G          BIT     #BIT2,@RXDB     ;INITIALIZE DONE?
005500  001774                          BEQ     30$             ;BR IF NOT
005502  004737  000000G                   JSR     PC,READ
005506  103404                          BCS     40$
005510  004737  000000G                   JSR     PC,EMPBF        ;EMPTY BUFFER FUNCTION
005514  004737  000000G                   JSR     PC,CMPBFA       ;VERIFY THE DATA
005520                          40$:
005520  005316                          DEC     (SP)            ;DONE?
005522  001335                          BNE     10$             ;BR IF NOT
005524  005726                          TST     (SP)+           ;RESTORE THE STACK POINTER
```

```
                        ; THIS TEST WILL LOAD THE HEADS AT TRACKS 0,76,1,75, ETC. THE TEST
                        ;WILL TRY TO LOAD THE HEADS AT THE SAME SPOT ON EACH TRACK.
                        ;FINALLY, A BAD BLOCK TEST WILL BE USED TO CHECK FOR MEDIA WEAR.
005526                  TEST22:
005526                          TEST    <TAP TEST>
005532                          STEST
005540  005037  000000G         CLR     SIDE2
005544  013746  000000G         MOV     TAPASS,-(SP)    ;NUMBER OF PASSES
005550  112737  000001  000000G MOVB    #1,SECT
005556                  10$:
005556  005037  000000G         CLR     TAPTRK
005562                  20$:
005562  004737  005720'         JSR     PC,60$          ;UNLOAD HEADS BEFORE SEEKING
005566  113737  000000G 000000G MOVB    TAPTRK,TRK
005574  004737  000000G         JSR     PC,READ         ;TO SEEK THE TRACK
005600  103410                  BCS     30$
005602  004737  005720'         JSR     PC,60$          ;DELAY
005606  005237  000000G         INC     IGNERR          ;IGNORE ERROR
005612  004737  000000G         JSR     PC,READ         ;TO LOAD THE HEADS
005616  005037  000000G         CLR     IGNERR
005622                  30$:
005622  004737  005720'         JSR     PC,60$          ;UNLOAD HEADS BEFORE SEEKING
005626  012746  000114          MOV     #76.,-(SP)      ;CALCULATE THE CORRESPONDING TRACK
005632  163716  000000G         SUB     TAPTRK,(SP)
005636  112637  000000G         MOVB    (SP)+,TRK
005642  004737  000000G         JSR     PC,READ         ;SEEK THE TRACK
005646  103410                  BCS     40$
005650  004737  005720'         JSR     PC,60$
005654  005237  000000G         INC     IGNERR
005660  004737  000000G         JSR     PC,READ         ;LOAD THE HEADS
005664  005037  000000G         CLR     IGNERR
005670                  40$:
005670  005237  000000G         INC     TAPTRK
005674  023727  000000G 000115  CMP     TAPTRK,#77.     ;DONE WITH A PASS?
005702  002727                  BLT     20$             ;BR IF NOT
005704  005316                  DEC     (SP)            ;DONE WITH TAPPING?
005706  001323                  BNE     10$             ;BR IF NOT
005710  005726                  TST     (SP)+           ;RESTORE STACK POINTER
005712  004737  000000G         JSR     PC,BLKCHK       ;BAD BLOCK CHECK
005716  000411                  BR      99$             ;EXIT
005720                  60$:
005720  013704  000000G         MOV     TAPDLH,R4
```

```
    43 005724  013705  0000006                 MOV     TAPDLL,R5
    44 005730                          65$:
    45 005730  005305                          DEC     R5
    46 005732  001376                          BNE     65$
    47 005734  005304                          DEC     R4
    48 005736  001374                          BNE     65$
    49 005740  000207                          RTS     PC
    50 005742                          99$:
```

```
                                ; THIS TEST WILL MOVE THE HEADS OF A SELECTED DRIVE TO
                                ;A SELECTED TRACK TO AID IN HEAD ALIGNMENT TESTING.
005742                          TEST23:
005742                                  TEST    <HEAD POSITIONING>
005746                                  STEST
005754                          5$:
005754                                  GTDRV
006162  103474                          BCS     99$             ;EXIT IF DRIVE NOT READY
006164  162700  000060                  SUB     #'0,R0          ;DRIVE NUMBER
006170  110037  000000G                 MOVB    R0,DRIVE
006174                          15$:
006174                                  REMARK  <ENTER TRACK # (0-76) OR INIT DRIVES (77) >
006260  004737  000000G                 JSR     PC,GTDEC        ;GET INPUT
006264  103002                          BCC     20$             ;BR IF NO ERROR
006266  005726                          TST     (SP)+
006270  000431                          BR      99$
006272                          20$:
006272  012600                          MOV     (SP)+,R0
006274  100737                          BMI     15$
006276  020027  000115                  CMP     R0,#77.         ;INIT DRIVES?
006302  001004                          BNE     25$             ;BR IF NOT
006304  012777  000000G 000000G         MOV     #BIT14,@RXCS    ;INIT
006312  000620                          BR      5$
006314                          25$:
006314  020027  000114                  CMP     R0,#76.
006320  003325                          BGT     15$
006322  005037  000000G                 CLR     SIDE2
006326  112737  000001  000000G         MOVB    #1,SECT
006334  110037  000000G                 MOVB    R0,TRK
006340  012737  000001  000000G         MOV     #1,IGNERR       ;IGNORE ERRORS
006346  004737  000000G                 JSR     PC,READ         ;POSITION HEADS
006352  000710                          BR      15$
006354                          99$:
000001                                  .END
```

SYMBOL TABLE

```
ABORT = ****** G    DELAY = ****** G    INTA  = ****** G    RANSEC= ****** G    STARTM= ****** G
ALGTSN= ****** G    DENS  = ****** G    IOERR = ****** G    RANSID= ****** G    STATUS= ****** G
AUTOM = ****** G    DIR   = ****** G    IPL   = ****** G    RANTRK= ****** G    STRBUF= ****** G
BDSEC = ****** G    DNWAIT= ****** G    LGLEC = ****** G    RANVAL= ****** G    STRK  = ****** G
BDTRK = ****** G    DRERR = ****** G    LGLECN= ****** G    RDEC  = ****** G    STRMEM= ****** G
BIT0  = ****** G    DRIVE = ****** G    LGLECZ= ****** G    READ  = ****** G    STRTUP= ****** G
BIT1  = ****** G    DRIVE0= ****** G    NCONTR= ****** G    READEC= ****** G    STTST = ****** G
BIT10 = ****** G    DRPTR = ****** G    NUM   = 000023      REG0  = ****** G    SYM   = 000001
BIT11 = ****** G    DSIDED= ****** G    NUM1  = 000023      REG1  = ****** G    TAPASS= ****** G
BIT12 = ****** G    EBUF  = ****** G    OPTION= ****** G    REG2  = ****** G    TAPDLH= ****** G
BIT13 = ****** G    ECODE = ****** G    OUTNUM= 000002      REG3  = ****** G    TAPDLL= ****** G
BIT14 = ****** G    ECTBPT= ****** G    PARM11= ****** G    REG4  = ****** G    TAPTRK= ****** G
BIT15 = ****** G    EMPBF = ****** G    PARM16= ****** G    REG5  = ****** G    TAPTSN= ****** G
BIT2  = ****** G    ENDTST= ****** G    PARM24= ****** G    REG6  = ****** G    TCNTNR= 000023 G
BIT3  = ****** G    EOP   = ****** G    PARM26= ****** G    RLINE = ****** G    TEST1   002004R
BIT4  = ****** G    ERBYCD= ****** G    PARM30= ****** G    RSEC  = ****** G    TEST10  003372R
BIT5  = ****** G    ERBYTK= ****** G    PARM31= ****** G    RSTS  = ****** G    TEST11  003422R
BIT6  = ****** G    ERROR = ****** G    PARM32= ****** G    RTCT  = ****** G    TEST12  004034R
BIT7  = ****** G    ERRVEC= ****** G    PARM33= ****** G    RXCS  = ****** G    TEST13  004156R
BIT8  = ****** G    ESEC  = ****** G    PARM34= ****** G    RXCS0 = ****** G    TEST14  004254R
BIT9  = ****** G    ESTAT = ****** G    PARM36= ****** G    RXCS1 = ****** G    TEST15  004360R
BLKCHK= ****** G    ETRK  = ****** G    PARM4 = ****** G    RXDB  = ****** G    TEST16  004604R
BUFFIL= ****** G    EXTEND= ****** G    PAR0  = ****** G    RXDB0 = ****** G    TEST17  004710R
BUFFIS= ****** G    FBUF  = ****** G    PAR1  = ****** G    RXDB1 = ****** G    TEST2   002230R
BUFLEN= ****** G    FILLBF= ****** G    PAR2  = ****** G    RXLEV = ****** G    TEST20  005154R
BUF1  = ****** G    FMTYPE= ****** G    PAR3  = ****** G    RXSEL = ****** G    TEST21  005400R
BUF2  = ****** G    FRMT  = ****** G    PAR4  = ****** G    RXVEC = ****** G    TEST22  005526R
B22FLG= ****** G    FUN   = ****** G    PAR7  = ****** G    RXVEC0= ****** G    TEST23  005742R
CCSBIT= ****** G    GO    = ****** G    PATRN = ****** G    RXVEC1= ****** G    TEST3   002306R
CDBBIT= ****** G    GTDEC = ****** G    PATRN1= ****** G    SAVE1 = ****** G    TEST4   002500R
CKDD  = ****** G    HDTRK = ****** G    PATRN2= ****** G    SAVE2 = ****** G    TEST5   002570R
CKINIT= ****** G    HEAD  = ****** G    PR0   = ****** G    SB    = ****** G    TEST6   003140R
CKTRK = ****** G    HOLDSC= ****** G    PR1   = ****** G    SCNTNR= 000023      TEST7   003342R
CLERRA= ****** G    IGNERR= ****** G    PR2   = ****** G    SECMAX= ****** G    TKS   = ****** G
CLERRZ= ****** G    IL    = ****** G    PR3   = ****** G    SECT  = ****** G    TMP1  = ****** G
CMPBF = ****** G    ILPTR = ****** G    PR4   = ****** G    SIDES = ****** G    TMP2  = ****** G
CMPBFA= ****** G    ILTBL = ****** G    PR5   = ****** G    SIDE2 = ****** G    TNMTAB  006430RG
CMPBFD= ****** G    IL1   = ****** G    PR6   = ****** G    SMEDIA= ****** G    TNUM  = 000024
CONTR = ****** G    INFLAG= ****** G    PR7   = ****** G    SOP   = ****** G    TOPMEM= ****** G
COUNTE= ****** G    INIT  = ****** G    RANDOM= ****** G    SR0   = ****** G    TRK   = ****** G
CRCVAL= ****** G    INITC0= ****** G    RANDT1= ****** G    SSEC  = ****** G    TRKMAX= ****** G
DDBBIT= ****** G    INITC1= ****** G    RANDT2= ****** G    START = ****** G    TRWAIT= ****** G
```